Redmine - Defect #31924

Paging misses some entries

2019-08-19 17:15 - Attie Grande

Status:	Closed		**Start date:**	
Priority:	Normal		**Due date:**	
Assignee:			**% Done:**	0%
Category:	REST API		**Estimated time:**	0.00 hour
Target version:				
Resolution:	Duplicate		**Affected version:**	4.0.4

Description

I've just been using the API to batch-update a large number of time entries (moving toggl ID from a comment into a custom field), and I noticed that a handful of records were not retrieved via `time_entries.json` (and thus weren't update), while a handful of others were returned multiple times.

- This happens for all page sizes `<= 99` that I've tried (`5`, `15`, `25`, `50`, `75`, `99`).
- Strangely for a page size `100` I get the expected output, though this could be fluke with where the page boundaries fall.
- Requests for page sizes `>= 100` return pages of `100` records.

I think It's also worth mentioning that by varying the page sizes, I see that records are not always returned in the same order...

For example, with pages of 5 records, `498` is returned twice:

- Page `n-1`: `505` `502` `503` `504` `498`
- Page `n`: `500` `498` `501` `497` `496`

Then, with pages of 25 records, `498` is returned once, with neighbours in a different order:

- Page `n`: `503` `502` `504` `500` `501` `498` `499` `497` `496` `493` `494`

I suspect that the indeterminate ordering is very related to this bug.

Note, I have also tried using the undocumented `page` parameter with no difference, as mentioned here:
https://www.redmine.org/issues/20479

Some example code:

```python
import json
import itertools
import requests

base_url = 'https://redmine.example.com'
headers = {
'X-Redmine-API-Key': 'YOUR_API_KEY'
}

params = {
'limit': 5
}

count = 0
entries = {}

for i in itertools.count(0, params['limit']):
params['offset'] = i

response = requests.get(f'{base_url}/time_entries.json', params=params, headers=headers)
  content = json.loads(response.content)
```

```
num = len(content['time_entries'])
  count += num

for e in content['time_entries']:
    e_id = e['id']
    if e_id in entries:
        raise Exception(f'ID {e_id} already exists!')

entries[e_id] = e

if content['offset'] + num >= content['total_count']:
    break

print(count)
```

I'm running Redmine using the [`redmine:4.0.4` docker image](https://hub.docker.com/_/redmine/)

Info tab:

```
Environment:
Redmine version                4.0.4.stable
Ruby version                   2.6.3-p62 (2019-04-16) [x86_64-linux]
Rails version                  5.2.3
Environment                    production
Database adapter               Mysql2
Mailer queue                   ActiveJob::QueueAdapters::AsyncAdapter
Mailer delivery                smtp
SCM:
Subversion                     1.10.4
Mercurial                      4.8.2
Bazaar                         2.8.0
Git                            2.20.1
Filesystem
Redmine plugins:
redmine_auth_gitlab            0.0.1
```

The database is MySQL using the [`mysql/mysql-server:8.0.17` docker image](https://hub.docker.com/_/mysql).

Related issues:

Is duplicate of Redmine - Defect #29581: Issues in paginated views may be los...	**Closed**

History

#1 - 2019-08-19 17:18 - Attie Grande

Gah, not markdown and I can't edit the original...

Also the example code was broken:

```
import json
import itertools
import requests

base_url = 'https://redmine.example.com'
headers = {
  'X-Redmine-API-Key': 'YOUR_API_KEY'
}

params = {
  'limit': 5
}

count = 0
entries = {}

for i in itertools.count(0, params['limit']):
```

```
    params['offset'] = i

    response = requests.get(f'{base_url}/time_entries.json', params=params, headers=headers)
    content = json.loads(response.content)

    num = len(content['time_entries'])
    count += num

    for e in content['time_entries']:
      e_id = e['id']

      print(e_id)

      if e_id in entries:
          raise Exception(f'ID {e_id} already exists!')

      entries[e_id] = e

    if content['offset'] + num >= content['total_count']:
      break

print(count)
print(len(entries))
```

#2 - 2019-08-20 04:48 - Go MAEDA

- Status changed from New to Needs feedback

Could you try the following patch extracted from [#29581](#)?

```
Index: app/models/time_entry_query.rb
===================================================================
--- app/models/time_entry_query.rb      (revision 18379)
+++ app/models/time_entry_query.rb      (working copy)
@@ -141,6 +141,7 @@
   def results_scope(options={})
     order_option = [group_by_sort_order, (options[:order] || sort_clause)].flatten.reject(&:blank?)

+    order_option << "#{TimeEntry.table_name}.id ASC"
     base_scope.
       order(order_option).
       joins(joins_for_order_statement(order_option.join(',')))
```

#3 - 2019-08-30 13:27 - Attie Grande

Hi,

Thanks for your reply, and sorry for my delay (it looks like your emails are going into spam by the way - you're getting SPF neutral).

I applied the full fix-29581-v3.patch, and it's resolved the issue for me - thanks very much!

Any idea when this patch will make it into a release? It looks like it's scheduled for v4.1.0?
I've incorporated it into my docker image for the moment.

Feel free to close this ticket as a duplicate.

Attie

#4 - 2019-08-30 15:46 - Go MAEDA

- Is duplicate of Defect #29581: Issues in paginated views may be lost because sorting criteria are not unique added

#5 - 2019-08-30 15:49 - Go MAEDA

- Status changed from Needs feedback to Closed

- Resolution set to Duplicate

Thank you for your feedback. I am happy to hear that the problem was fixed by the patch.

I think that Redmine 4.1.0 should be released soon, but I don't know the accurate date.